ATTR Syntax: Attr filename [permissions] Usage : Examine or
change the security permissions of a file Opts: -perm = turn off
specified permission perm= turn on specified permission -a =
inhibit rms : d - directory file
s - nc to owner w - write permit
to own r - read permit to public
pw - te permit to public BACKUP
Syntax ge : Copies all data from
one de ead error occurs s =
single writes BASIC09 Syntax :
Basic0 ge BUILD Syntax: Build
filenar s from standard input
CHD Sy nge working directory to
specifi > Usage: Change execution

AUSTRALIAN OS9 NEWSLETTER

directory to specified path CMP Syntax: Cmp filename1 filename2
Usage : File comparison utility COBBLER Syntax: Cobbler devname
Usage : Creates OS-9 bootstrap file from current boot CONFIG
Syntax: Config Usage : Create custom boots and system disks COPY
Syntax data from
one fil E Syntax :
Date [t Opts: t =
specify ame> Usage
: Check directory
for wor sters -m
= save of unused
cluster nly -o =
print <devname>
}<devn : Del [-x]
filenam s : -x =
delete : Deldir
directo yntax: Dir
[e] [x] the file
names y x=print
executi] Usage :

EDITOR :
Gordon Bentzen
8 Odin Street
SUNNYBANK Qld 4109

(07) 345-5141

Display s converted characters to standard output DSAVE Syntax
: Dsave [-opts] [dev] [pathname] Usage : Generates procedure file
to copy all files in a directory system Opts : -b make a system
disk by using OS9boot if present -b=<path> = make system disk
using path as source -i = indent for directory levels -l = do not
process b makdir
command o num K
ECHO Syn tandard
output ED oriented
text edito s text

MAY 1989

error messages for given error numbers EX Syntax: ex <modname>
Usage: Chain to the given module FORMAT Syntax : Format
<devname> Usage : Initializes an OS-9 diskette Opts ; R - Ready L
- Logical format only "disk name" 1/2 number of sides 'No of

AUSTRALIAN OS9 NEWSLETTER
Newsletter of the National OS9 User Group

EDITOR : Gordon Bentzen

HELPERS : Bob Devries and Don Berrie

SUPPORT : Brisbane OS9 Level 2 User Group.

Welcome to yet another edition of our newsletter which is more of the same, more OS-9 that is.

We have received several requests for public domain software during the last month; requests which I believe have all been filled. I had a call from one member who had received his disks by return mail only to find that I had formatted one of his disks and apparently forgotten to copy files to it. Please let me know if I have slipped up in any way with material you have requested.

Please, please send us formatted disks when requesting copies of the public domain disks. Firstly, this makes the job of copying just a little easier for us, but more importantly you will be sure of getting back the disk in the format that you need and you should not have any difficulty in reading the files.

All our public domain library is stored on 80 track double sided disks, so if you can handle this format, this is the easiest for us and you will get more files (generally) for the $2.00 per disk copy charge. Also remember to include postage stamps for return postage.

We would like to thank those members who have contacted us with their OS-9 questions. These questions help us to determine what your needs are, and to hopefully make this newsletter a little more interesting. You know we are flying blind a good deal of the time in the area of selecting articles that readers want to see. The membership of the National OS9 User Group includes those with a good deal of experience and those new to OS9, so we are trying to present a newsletter that will have some interest for everybody. We do have a couple of questions for which we do not have an answer at this point. These relate to OS9 application programmes such as PhantomGraph, Home Publisher and the like. Please don't give up on us yet, our research is still in progress.

The software reviews that we have included in these pages in past editions were apparently well received by a number of members, so we would be glad to print your comments on any OS9 software package that you use. Come on guys, share with us all those little do's and don'ts that you have learnt the hard way.

This newsletter is produced using "Stylograph" by Great Plains Computer Co. Idaho U.S.A. running on a CoCo3 512k. Stylo does quite a good job, but does do some strange things with big files at times. Can anybody comment on this?

In this edition we have included a listing of all current members of the User Group for your information. We have not included member's telephone numbers as not all members may want their number published. The information presented will however, allow you to make initial contact by mail if you wish. You may just be suprised to learn that you are not alone as an OS9 user in your area.

Part 3 of the database in C by Bob Devries is included, as is some further comments from me on the OS9 operating system. Don Berrie may have a little surprise for us, so read on.

Ross Pratt from Cooma has submitted the source for his "hxd" hexadecimal dump together with his comments on this handy utility. Many thanks Ross.

Until next month, happy OS9ing.

Regards,
Gordon Bentzen

Hexadecimal and ASCII dump Programme
by Ross Pratt.

The reason I wrote this Hexadecimal and Ascii dump programme was for a job I had to do at work and I did not like the format of the od unix utility, so I decided to write something with the same format as the os9 dump utility.

The programme which I have called hxd can have any number of arguments (files) to dump out. It can also start the dump at any point in the files. Hxd must have at least one command line argument. The first thing it checks for argc to be less than 2, if true it prints the format on the screen and then exits. Next it checks for the offset option, if this is true it again checks for at least two arguments. If all is ok than it gets the offset and places it into num. The while statement checks the number of arguments and then decrements it by one, next it attempts to open the file of that name. If it fails a message is printed on the screen and the programme exits. After a successful call to open the file, the name of the file is printed on standard out and then the function dump is called. This function prints out a block of 16 characters in hexadecimal and then in ascii. First it calls pflinit to tell the compiler that it will be printing out a long integer, then it prints out the heading. The file pointer is set at the offset (zero or whatever). The first for loop checks for the EOF character and increments j which is printed at the beginning of each line. The next for loop steps along the file 16 characters and places each character in line, at the same time line is spaced out as it is printed. The if statement checks the value of k and if it is less than 16 the following for loop fills out the line with spaces, as it will be the last line. Next the space between the hexadecimal and the ascii section of the line is printed. The last for loop prints out the ascii section. The if statement checks that each character is an ascii character and if not, then a dot is put in it's place and the character printed. A new line is printed to start over. The last if statement increments i by 1 and checks for it being equal to 16, if true 2 lines and another heading are printed also i is set to zero. At the EOF character the function returns, 2 lines printed and the file closed. The while loop again check for another argument and if true the process repeats over.

I now use this programme instead of dump as I can offset the starting point of the dump and it spreads the hex characters out and this is easier to read. I hope you readers enjoy this programme. Now here's the source..........

```
/* Programme    Hxd -- (Hexadecimal Dump)
 * Author       Ross Pratt
 *              31 Campbell St.
 *              Cooma. NSW.
 * This programme is to be used to dump out all the characters in each file
 * in it's argument list, the format is hexadecimal and ascii.
 */
#include        <stdio.h>
#include        <ctype.h>

main(argc, argv)
int     argc;
char    *argv[];

{
FILE    *fopen(), *inf;
char    anum[7];
long    num = 0, atol();
int     i = 2, j = 0, a = 1;

if(argc < 2)    /* check for at least 1 argument */
        format();
else
        {       /* check if Offset parameter */
        if(argv[i][0] == '-' && argv[i][1] == 'o')
                if(argc < 3)    /* is there enough arguments */
                        {
                        format();
                        exit(0);
                        }
                else
                        {       /* get the amount of offset */
                        while(argv[i][i] != '\0')
                                anum[j++] = argv[i][i++];
                        num = atol(anum);
                        a++;
                        --argc;
                        }
        while(argc-- > 1)
                {       /* open file(s) to dump */
                if((inf = fopen(argv[a], "r")) == 0)
                        {       /* could not find this filename */
                        fprintf(stderr,
                 "Unable to Open %s  --- Programme exiting Now!!\n", argv[a]);
                        exit(0);
                        }
                        /* Print out the File Named */
                printf("         Dump of %s\n\n", argv[a++]);
                dump(inf, num);
                printf("\n\n");
                fclose(inf);
                }
        }
}
dump(fp, j)       /* dump - the function that does all the work */
FILE    *fp;
long    j;

{
int     k = 0, i = 0, l, m;
int     line[16];

pflinit();
heading();
fseek(fp, j, 0);
for(; line[k] != EOF; ++j)
        {
        printf("%05lx    ", j);
        for(k = 0; k < 16; ++k)
                {
                if((line[k] = getc(fp)) == EOF)
                        break;
                printf(" %02x", line[k]);
                }
        if(k < 16)
                for(m = k; m < 16; ++m)
                        printf("   ");
        printf("       ");
        for(l = 0; line[l]!= EOF && l < 16; ++l)
                {
                if(line[l] < 32 || line[l] > 127)
```

```
                        line[l] = 46;
                printf("%c", line[l]);
                }
        printf("\n");
        if(++i == 16)
                {
                printf("\n\n");
                heading();
                i = 0;
                }
        }
}
heading()       /* Heading - prints out the Heading at the top of each Block */
{
printf("Address   00 01 02 03 04 05 06 07 08 09 0A 0B 0C 0D 0E 0F      ");
printf("0123456789ABCDEF\n");
printf("-------------------------------------------------------------------");
printf("----------------\n");
}
format()        /* Format - print the correct format when needed */
{
fprintf(stderr,"Incorrect argument List\nhxd [-ooffset] filename[ filename]\n");
}
```

oooooooooo0000000000oooooooooo

The OS9 Operating System

In the last two editions we touched on OS9 module types, the kernel, bootstrapping the system and how to merge any selected user modules with the shell module. It does seem that at least some readers of the newsletter are interested in the comments presented. So if you can stand it, I will prattle on (to borrow Bob's description of my efforts) a bit more.

It is hard to know just where to go from here, but perhaps a few comments on getting the most out of OS9 would be in order, especially for the beginners. This also follows a theme set by Rosko in his well presented articles on Microware C.

I will comment on OS9 running on the CoCo systems as this probably covers most people new to OS9.

Firstly, the hardware requirements that are desirable lead us to thoughts of a hard disk system, however such a system is probably not for everybody for a number of reasons, not the least of which is the price tag. We do need to consider just how far the budget will stretch or just what sort of investment we are willing to make. The decision we each make will also be influenced by our own reasons for running OS9, or for computing at all.

Well, I am one of those who has not as yet installed a hard disk system and can only look on with envy at those hard diskers, a la Don Berrie and many others in our group.

O.K. so all this doesn't help you very much. A very workable and useful system can be achieved without a hard disk.
Firstly you will definitely need two (2) floppy drives. Although OS9 can be run on a single drive system, it is nothing less than painful, especially if that single drive is only 35 or 40 track single sided.

The OS9 operating system uses disk access a lot. So if you do want to run OS9 and have a single drive system, then do yourself a favour and invest in a second drive. The frustration you will save yourself will be well worth the investment. I am running three (3) 80 track double sided drives, which although not quite my plan, have proven to be a great advantage. The number of disk changes I have to do is now greatly reduced, as I can boot from a system disk in drive 0, run applications from drive 1, and still have another drive for data disk or whatever. It is great to have a full CMDS directory sitting in drive 0, and be able to 'dsave' from /d1 to /d2.

Now if you would like to see your favorite programme running even faster than it would on a hard disk system, a ram disk will speed things up no end. This is of particular advantage with such programmes as "Profile" or other databases. There is a public domain driver "rammer" and descriptor "r0" which can be added to the OS9Boot. The CoCo3 system will need to have 512k to use this effectively. We have been using R0 to simulate a 35tr single sided disk, which allows a 'backup' of a standard disk, and is a quick way of loading programmes or data files to the ram disk. The use of a ram disk for data of course has some risks as this is hardly a permanent record. When data processing is completed the updated files can be transferred to a floppy by a file copy or backup. The data on the ram disk r0 can be recovered even after a re-boot of OS9. If "iniz r0" is the first command in your startup file and "format /r0" is NOT included, the data previously stored will still be there. A simple 'dir /r0' will confirm this. Thanks to Don Berrie for this discovery. Don also provided the source code for a "r0" descriptor that allows it to be used with D.P.Johnson's "SDisk3"; refer to our July '88 newsletter.

Just a brief aside before I move off disk drives. As I said, my system was not planned around 3 x 80 track drives. Two drives worked just fine apart from the fact that the 80 track drives could not conveniently read a standard 35tr. RS Dos disk. So, as many CoCo owners have been using TEAC FD55F 80 tr drives which could easily be made to simulate 40tr mode by simply switching out a particular resistor on the drive board, I decided to take the plunge and order such a drive. TEAC have revised the FD55F to such an extent that we have not been able to find a method of the desired mode switching. Has anybody out there found a simple solution to this problem?

If you are using a CoCo 1 or CoCo 2 you will of course be running OS9 Level 1 with 64k of memory. Beyond this you can add extra memory which is usable only as a ram disk. This does add some advantage to level 1, but unless you have already made this modification, I would suggest that the purchase of a CoCo 3 or perhaps something else would be a better way to go.

If you already have a CoCo 3 and have not added a 512k upgrade, then may I suggest that you do. The Microware OS9 level 2 distributed by Tandy for the CoCo 3 almost demands this 512k. I say almost because it will run on a 128k computer, but you will very quickly find that multitasking is extremely limited.

In summary then, two disk drives are a must, 512k on a CoCo 3 is also a must. Now you will be able to get close to running OS9 the way Microware intended it. You will soon find however, that the CoCo hardware, in particular the operation of the disk controller, will severely limit multitasking. The Tandy disk controller interface uses interrupts of the processor which prevents any other processing to take place whilst a disk drive is being accessed. The result is that you find yourself waiting for each disk access to complete before you can do anything. The OS9 clock module will not keep correct time for this very reason.

Now there is an answer to this problem. It does require the parting with a few more dollars though. I have recently added a 'no-halt' controller to my CoCo3 system which has resulted in such an improvement that I can only describe it as MAGIC. I have the "Disto" controller manufactured and distributed by CRC COMPUTERS Montreal, Canada.
This controller works on all CoCos 1, 2 & 3 with or without the Multi-Pak interface. The controller comes with special device drivers needed to utilize the special buffered mode. This uses memory installed in the controller so that all sectors being written to or read from the disk will be done via a sector buffer. The result is that the 6809 is not halted between bytes, and you are able to fly along without waiting for the CPU to allow reading of the keyboard or anything else for that matter. The manual does warn that you should not do anything else whilst formatting a disk because 'format' takes up almost all the processor time. Well warnings are there to be ignored aren't they. I have carried on running other processes whilst in the middle of a format without problems, although things do s l o w down noticeably.

I have been sidetracked this month and not covered much of the operating system as such, but I do hope that my comments on the desirable hardware will help some of you to make your venture into OS9 a little more enjoyable. Next month I will try to return to the subject of the OS9 operating system. Perhaps a short note to us with a suggestion on what you need to know will get me back on the track.

Gordon Bentzen

ooooooooooOOOOOOOOOOoooooooooo

Motor On modifications for FD502 disk drives
A further note by Bob Devries.

In last months issue I mentioned the jumper change necessary for the Tandy FD series disk drive, to allow the motor-on signal to turn on the drive motor instead of the select signal, which would otherwise create problems especially under OS9. I have since found out that the change I described was for the FD501 drive only.
Here is the modification for the FD502 drive. The main control board has a jumper marked '5' which needs to be moved to the alternate position also marked '5'. Unfortunately, the board needs to be removed to change the jumper. To do this, remove the three plugs near the 34 way connector, being careful with the two flimsy plastic ones. The three conductor plactic one needs to be released by prising up the white plastic collar that holds it, and then removing the lead. Undo the three screws and remove the board, not that it is still held by the motor connexions. Remove that connector also.
Locate the jumper marked '5' next to the 34 way connector, and remove the link. Solder a new link into the alternate position, and you're all done. Re-assemble the board to the drive, and re-install the connectors.

oooooooooo0000000000ooooooooooo

PROBLEMS WITH DSAVE

I have two single sided disk drives and at one time I was running only 35 tracks with a slow stepping rate. After talking to fellow OS9'ers I realised that my drives can cope with 40 tracks and 6 millisecond stepping rate. Bewdy I thought, more disk space.

I got hold of a 40 track boot disk through the local user group and promptly set out to reformat all my OS9 disks and transfer all the files onto the newly formatted disks. Easy - just type backup /d0 /d1 and away you go. Like fun. The error message comes up with "disks not formatted the same!". Well of course not, thats the whole idea! I was now in a dilemma - what can I do. I could try copying one file at a time but that would take from now to eternity to accomplish! I know what to do; I'll go BACK to RS-DOS and use the basic command "BACKUP" but that didn't work too well either.

I rang Bob Devries and asked him what I should do. He said to use DSAVE. "What is DSAVE?" I enquired. The answer is what I wanted to hear "Well DSAVE is a command in your CMDS directory and is specifically designed to transfer files from differently formatted disks." "THANK YOU BOB" was my reaction and away I went to use my new found knowledge.

Into drive 0 goes my 35 track system disk and into drive 1 goes the newly formatted 40 track disk. Then I followed the instructions given to me. DSAVE /d0 /d1 ! SHELL . What a bewdy! Its working! My files are being transferred all by them selves. What's that! ERROR 244! Not to worry only one out of a hundred files. Oops not again - another error ERROR 243 Oh no not another! Well DSAVE seemed like a good idea at the time. I know what to do. Just write down the files that DSAVE didn't copy properly and copy them in 'long hand' later. After copying ten files across I finally had a system disk on 40 tracks.

Proud of my new aquisition I went to work on it. First thing I did was to boot up with it. Everything was fine until the startup file tried to read the PRINTERR file and came up with the dreaded ERROR message. ERROR 214 NO PERMISSION. What's all that about! PRINTERR with no permission. Bull! Well lets use ATTR PRINTERR and see. ds_wr_wr is what came up. No can't be - PRINTERR is not a directory. Oh well its easy to fix. DELDIR /d0/CMDS/printerr is the logical answer. Guess what - it didn't work! I tried to use ATTR PRINTERR -D but that didn't work either. Nothing I could do would allow me to change the file or delete the file! All I could do was to rename it. SO I did - I called it RINTERR, then I copied the file into the command directory from my backup disk. I took a closer look at other files and found that several other files were also "directories". I could not delete them either.

This problem had me bamboozled for a long time. I would ask people for help but no-one could help me. Now what can I do?

Well it so happened that one of the ladies at work had an old full height disk drive (bare) that her hubby couldn't get to work. She gave it to me to give to my boys for final destruction. I didn't give it to the boys but took it to a friend's place to try to resurrect it. While we worked on it quietly, I noticed that when he addressed drive 0 the motor on drive 1 would start up. How come? It seems that OS9 requires it as it does not allow time for the drives to get up to speed. When using BACKUP or (wait for it) DSAVE it may corrupt the disk being written to because it is trying to write to a disk that is not up to speed yet! Sounds too good to be true!

Off home I went to once again apply my knowledge. Off came the diskdrive cover and in went the fingers. Would you believe it. My TANDY drive is not latched with the Chinon drive. So I found out how to latch it. On the circuit board is a jumper link (similar to the one to select the drive number) with another pin next to it. The extra pin had been cut off. I soldered another pin on and relocated the jumper link. Now when either drive is addressed the other drive motor starts up. Fantastic!

Quick. Out with OS9 system disk. Try DSAVE. HURRY! Well it seems to be working but don't get your hopes up too high. DSAVE went right through the system disk WITHOUT any errors! Amazing ain't it. I can now run SDISK and a couple of other programmes that address two drives without any problems.

Now I am a happy little OS9'er once again and keen to use commands such as DSAVE as much as possible. So to anyone else who is having trouble with DSAVE - just check that the your drive motors are latched.

Peter Barendrecht.

oooooooooo0000000000oooooooooo

A Database in C.
By Bob Devries

This is part three of my database programme in C. The two functions printed here draw the data mask on the screen, and put the current record into the blanks.

The function scrnmask() only prints to certain locations on the screen and clears the screen of a previous data record. The function scrndata() reads the a record, whose number is given in the recno parameter passed to it, and prints each field in its correct location on the screen.

Here then is this month's code.

```
scrnmask()
/* scrnmask() clears the screen and draws the mask of the data boxes with */
/* their titles */
{

cursor(5,5);      /* position cursor */
printf("Surname [                    ]");      /* print field contents */
cursor(40,5);
printf("Firstname [                   ]");
cursor(5,7);
printf("Street [                     ]");
cursor(5,8);
printf("City [                     ]");
cursor(5,9);
printf("State [   ]");
cursor(5,10);
printf("Postcode [    ]");
printf("\n\n\n");
}


scrndata(recno)
int recno;
```

```
/* scrndata() reads the current recno from the file and prints it on the */
/* screen in the alloted places */
{
        if (recno < 1)       /* test to see if you've gone too far back */
                recno = 1;  /* and set to one if so */
        fseek(fp,(long)((recno-1)*sizeof(mail)),0);    /* goto record number */
        fread(&mail,sizeof(mail),1,fp);       /* read the data in */
        scrnmask();                           /* clear the screen of old data */
        cursor(14,5);        /* position cursor */
        printf("%s",mail.surname);      /* print the field contents */
        cursor(51,5);
        printf("%s",mail.firstname);
        cursor(13,7);
        printf("%s",mail.street);
        cursor(11,8);
        printf("%s",mail.city);
        cursor(12,9);
        printf("%s",mail.state);
        cursor(15,10);
        printf("%4d",mail.postcode);
}
```

oooooooooo0000000000oooooooooo

SETTING THE ENVIRONMENT FOR YOUR C COMPILER

The following short Basic09 programme arose from a conversation with Bob Devries. We were discussing problems relating to the Microware C Compiler, and it's lack of any method to automatically include graphics, and other library support. This problem is caused by the fact that cc1 (or the patched "cc2" version .. see September 1988 Newsletter) writes a procedure file, c.com, which handles all of the sequential steps of preparation, compiling, optomizing, assembly and linkage. Unless you physically edit this file to include other library resources, the linker will generate "unresolved reference" error messages when you try to call functions which are not in the standard library.

We figured that the best way to achieve this was to use the -r option with the existing compiler, and then add the necessary library options via a Basic09 procedure. As an added feature, we thought that it would be nice to include full pathnames to the relevant directories. This will make it especially useful for hard disk users. One thing you will have to remember, however, is that the standard compiler looks in the directory /d1/DEFS to locate its #include files, when they are enclosed in <>'s, as in #include <stdio.h>. If you plan compile programs outside those specific directories, you will need to provide full pathnames in your source code, and enclose them in parenthesis. eg #include "/H0/COMPILER/DEFS/stdio.h". I also have included sufficient error trapping so that if there is a problem, you won't be left with myriad overlay windows hanging around.

At about the same time, I had also been fiddling with some interesting screen manipulation techniques, so I decided to include an exploding window effect which is well worthwhile having a look at, even if you are not interested in the environment setting part of the programme.

The programme also attempts to save the pathnames (again for hard disk and 80 track drive users) to a directory named SYS in the root directory of the device containing the current data directory. Boy, what a mouthful. But think about it. The whole thing is designed to run in an 80 column window, and will need the gfx2 graphics support module in memory, or in the execution directory. It also makes a couple of other system calls, so it will also need to have access to them. You'll see the SHELL calls in the source code. There is just one caveat. Remember, this was an extremely rough and quick exercise, and therefore I cannot gaurantee that there are no hidden errors or other hiden lurkers. There is at least a framework for some pretty neat improvements. I look forward to seeing some modifications from within our membership.

If you would like to discuss the programme, I will be only too happy to talk to you. Please contact me on (07) 375-3236.

Cheers   Don Berrie.

```
   PROCEDURE envmaker
   RUN explode
   DIM fpath,flag:BYTE
   DIM cfile,rfile:STRING[30]
   DIM keypress:STRING[3]
   DIM errno:INTEGER
   DIM libpath,cmdspath,sourcepath:STRING[30]
   PRINT
   PRINT "Searching for ....../SYS/ccomp.env"
   PRINT
   ON ERROR GOTO 10
   OPEN #fpath,"........../SYS/ccomp.env":READ
   ON ERROR
   PRINT
   GOTO 15
10 errno=ERR
   ON ERROR
   IF errno=216 THEN
    PRINT "Environment directory/file not found"
   ENDIF
   PRINT
   PRINT "Input pathname for library files (eg. /H0/USR/LIB... ) :";
   INPUT libpath
   PRINT
   PRINT "Input pathname for Compiler CMDS directory :";
   INPUT cmdspath
   PRINT "Input pathname for Compiler source directory :";
   INPUT sourcepath
   PRINT "Create environment file (Y/N) :";
   INPUT keypress
   IF keypress="Y" OR keypress="y" THEN
    CREATE #fpath,"........../SYS/ccomp.env":WRITE
    WRITE #fpath,libpath
    WRITE #fpath,cmdspath
    WRITE #fpath,sourcepath
   ENDIF
   GOTO 18
15 READ #fpath,libpath
   READ #fpath,cmdspath
   READ #fpath,sourcepath
18 CHD sourcepath
   CHX cmdspath
   PRINT "Input filename to compile :";
   ON ERROR GOTO 100
   INPUT cfile
   PRINT
   PRINT "Is graphics support required (Y/N) :";
 20 INPUT keypress
   keypress=LEFT$(keypress,1)
   IF keypress="Y" OR keypress="y" THEN
   flag=1
   GOTO 25
   ENDIF
   IF keypress<>"N" OR keypress<>"n" THEN
   GOTO 20
   ENDIF
   flag=0
 25 SHELL "cc2 "+cfile+" -r"
   rfile=LEFT$(cfile,LEN(cfile)-2)+".r"
   PRINT "rlink:"
   IF flag=0 THEN
   SHELL "rlink "+libpath+"/cstart.r "+rfile+" -o="+LEFT$(cfile,LEN(cfile)-2)+" -l="+libpath+"/clib.l"
   ELSE
   SHELL  "rlink  "+libpath+"/cstart.r  "+rfile+"  -o="+LEFT$(cfile,LEN(cfile)-2)+"  -l="+libpath+"/clib.l -
l="+libpath+"/cgfx.l -l="+libpath+"/sys.l"
   ENDIF
   SHELL "del c.com"
   SHELL "del "+rfile
   RUN unexplode
   PRINT "Program successfully compiled"
   CLOSE #fpath
   END
100 RUN unexplode
   CLOSE #fpath
   ON ERROR
   PRINT "ERROR - Program not compiled"
   SHELL "del "+rfile
   SHELL "del c.com"
   END


   PROCEDURE explode
   RUN gfx2("owset",1,39,11,2,2,5,5)
   RUN gfx2("owset",1,29,8,22,8,5,5)
   RUN gfx2("owset",1,19,5,42,14,5,5)
   RUN gfx2("owset",1,9,2,62,20,5,5)
   RUN gfx2("owset",1,0,0,80,24,2,5)
   RUN gfx2("owset",1,2,1,76,22,2,0)



   PROCEDURE unexplode
   DIM i:INTEGER
   FOR i=1 TO 6
   RUN gfx2("owend")
   NEXT i
```

AMBROSI	JULES	172 OGILVIE STREET	ESSENDON	VIC 3040	CoCo3
BARENDRECHT	PETER	181 WHEELER CRES.	WANNIASSA	ACT 2903	CoCo 2
BENTZEN	GORDON	8 ODIN STREET	SUNNYBANK	QLD 4109	COCO 3
BERRIE	DON	25 IRWIN TERRACE,	OXLEY	QLD 4075	COCO/ATARI
BISSELING	FRED	P.O.BOX 5447	BOROKO. N.C.D.	PNG	CoCo 2&3
BLANDFORD	George W	27 CANBERRA STREET	MOE	VIC 3825	
BRODIE	MICHAEL	P.O. BOX 109	HUNTINGDALE.	VIC 3166	
BROWN	ROHAN	75 PEMBROKE ROAD	MOOROOLBARK	VIC 3138	CoCo3
CLARKE	IAN	12 MARFAYLEY STREET	SALISBURY	QLD 4107	CoCo3
COOMBS	RON	25 MORILLA AVENUE	CARLINGFORD	NSW 2118	IBM CLONE
DEAN	JAMES	P.O. BOX 549	STH WINDSOR	NSW 2756	CoCo 2&3
DEVRIES	BOB	21 VIRGO STREET,	INALA	QLD 4077	COCO/AMIGA
EATON	DAVID	20 GREGSON PLACE,	CURTIN.	ACT 2605	ATARI
EDWARDS	PETER	40 DAVISON STREET,	MITCHAM.	VIC 3132	COCO3
ESKILDSEN	OLE	11 MONARCH STREET,	KINGSTON	QLD 4114	COCO3
FRANCIS	GEORGE	31 DONALD STREET	MORWELL	VIC 3840	CoCo3
FROST	PHIL. A.	25 CHEETHAM STREET,	KALGOORLIE.	WA 6430	COCO3
HARRIS	MICHAEL	P.O. BOX 25	BELMORE	NSW 2192	COCO3 512K
JACQUET	J.P.	27 HAMPTON STREET.	DURACK	QLD 4077	CoCo3
JENKINSON	CEC	49 HUTHWAITE ST.	WAGGA WAGGA	NSW 2650	CoCo 2
KEWLEY	DOUGLAS	2 DRYSDALE AVENUE,	TEE TREE GULLY	SA 5091	COCO1,2,3
LOWE	MIKE	441 MARANGAROO DRV.	ALEXANDER HIGHTS	WA 6064	CoCo3
MACLEOD	IAIN	150 COOLIBAH DRIVE	GREENWOOD	WA 6024	
MALONEY	Paul	8 GLOUCESTER STREET	JUNEE	NSW 2663	CoCo3
MANNING	PETER	7 CALNON STREET,	BASSENDEAN.	WA 6054	CoCo3
MARENTES	NICKOLAS	61 CREMIN STREET	UPPER MT.GRAVATT	QLD 4122	COCO3
MARTIN	TED	P.O. BOX 56	ROSNY PARK	TAS 7018	CoCo1,2,3
MAY	PETER	11 HUTTON STREET	CLAYFIELD	QLD 4011	CoCo3
McKAY	ROSS	56A CORNELIA ROAD	TOONGABBIE	NSW 2146	CoCo3
McLINTOCK	GEORGE	7 LOGAN STREET,	NARRABUNDAH.	ACT 2604	CoCo 1&3
McMASTER	BRAD	119 WILLOUGHBY ROAD	CROWS NEST	NSW 2065	VARIOUS
MIKULSKI	JOHN	56 KYOOMA STREET	TAMWORTH	NSW 2340	CoCo3
O'DONNELL	BILL	47/2 FRANCIS STREET	ARTARMON.	NSW 2064	
OBLAK	GERD	58 ELIZABETH PARADE	LANE COVE	NSW 2066	CoCo3
OOSTERBEEK	ROBIN	P.O. BOX 1123	DANDENONG	VIC 3175	CoCo3
PATRICK	WAYNE	12 O'CONNELL ST.	GYMPIE	QLD 4570	CoCo 2
PEARCE	W.LEIGH	47 ALLENBY AVENUE,	RESERVOIR.	VIC 3073	CoCo II
PRATT	ROSS	31 CAMPBELL STREET,	COOMA	NSW 2630	CoCo3
REID	THEO	35 AVONMORE AVE.	PORTLAND	VIC 3305	
RUPE	RON	18 DENSTON WY,	GIRRAWHEEN	WA 6064	CoCo3
SCHIPPLOCK	KEVIN	19 CELTIS STREET	ACACIA RIDGE	QLD 4110	CoCo3 x2
SIDEBOTTOM	Barry	31 NOTRE DAME DVE.	SUNBURY	VIC 3429	SOLD CoCo
SIMPSON	ANDREW	6 S.C.KING COURT	COLLINGWOOD PARK	QLD 4301	512K COCO3
SINGER	MAURICE	1 ATKA STREET	TREGEAR	NSW 2770	CoCo3
SKEBE	JEFF	92 BYNYA ROAD	PALM BEACH	NSW 2108	LYNSTAN
SLADE	ARTHUR F	P.O. BOX 516	SEVEN HILLS	NSW 2147	CoCo3
SPOTSWOOD	GARY	10 DOUCH STREET	WILLIAMSTOWN.	VIC 3016	CoCo2 B
TARVIN	DIGBY	P.O. BOX 498	RANDWICK	NSW 2031	AMIGA
UNSWORTH	ROB	20 SALISBURY ROAD,	IPSWICH	QLD 4305	CoCo3
USHER	JOHN	47 POLARIS AVE.	KINGSTON	QLD 4114	512K COCO3
WAGNITZ	KEN	2 DEPINDO AVE	EDEN HILLS	SA 5050	COCO3
WHITE	GREG	2/23 LOWER PLENTY RD	ROSANNA.	VIC 3084	CoCo3
WRIGHT	RON	2 IRENE CRT.	CHELSEA	VIC 3196	COCO3
WRIGHT	SEAN	8 SWORDFISH AVENUE,	RABY.	NSW 2566	CoCo3 x 2
Total Members:	54				